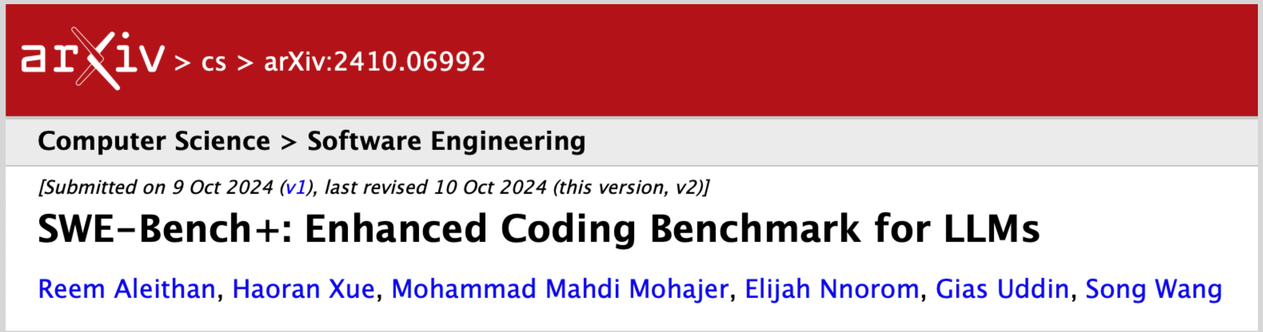
challenges of AI-based software development

Daniel Jackson & Eagon Meng · MIT CSAIL/EECS · April 2026

are LLMs good coders?

a benchmark and its analysis

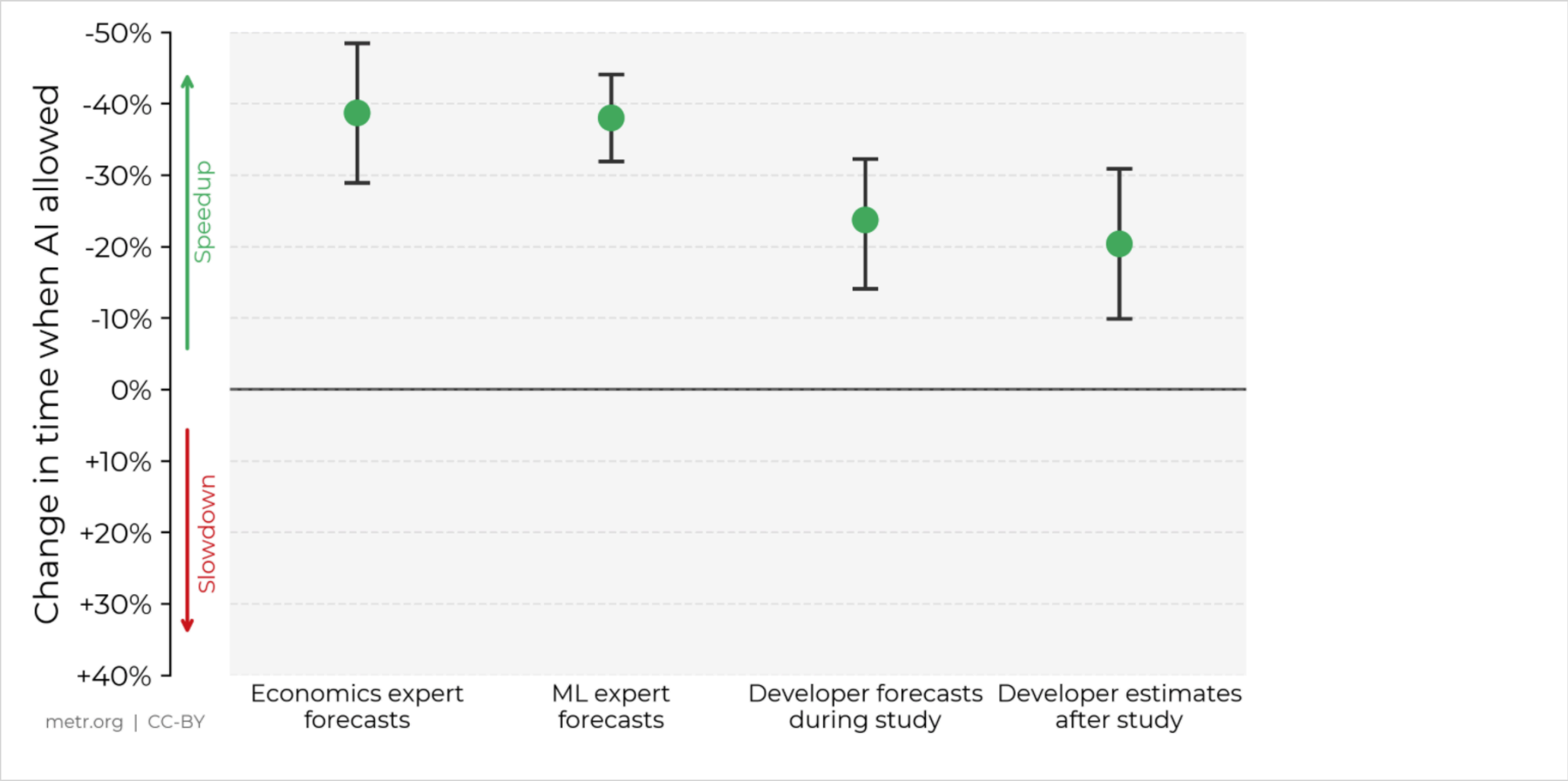
a benchmark for realistic coding problems (2024)
2,294 issue/pull request pairs from 12 Python repos
best LLM resolves 65% of issues



follow-up study at York University (2025)
33% of good patches "cheated": code appears in issue
31% of patches deemed correct by incomplete tests
94% issues were present before training cutoff
with all this, resolution rate for GPT-4 falls to 0.55%

in short: LLM-based coding assistants
often suggest code that doesn't work
and breaks existing functionality

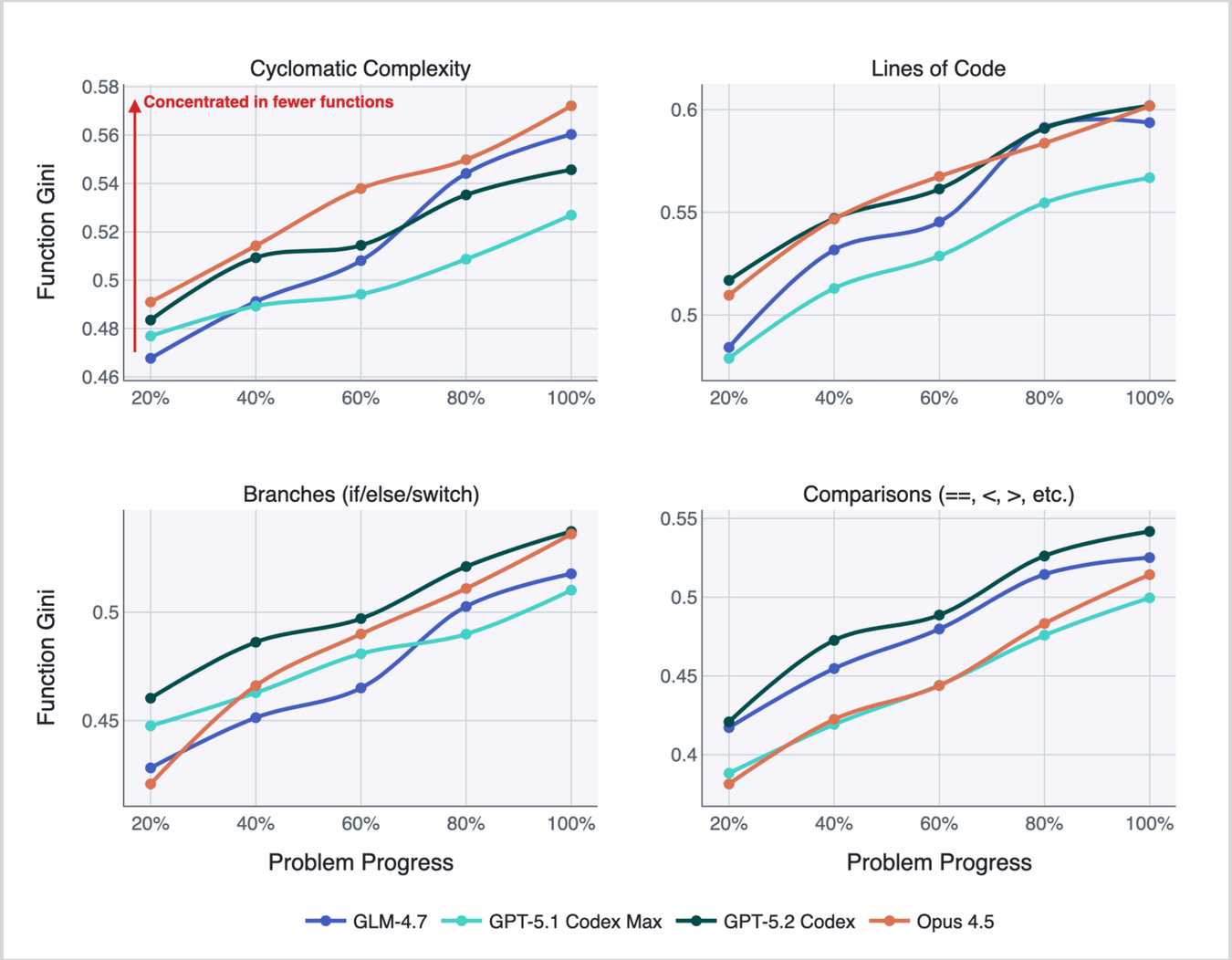
how much does AI speedup skilled developers on real codebases?



METR study (10 July 2025)
Randomized control trial
16 developers on 246 tasks

Developers qualitatively note LLM tooling performs worse in more complex environments. One developer says “it also **made some weird changes in other parts of the code** that cost me time to find and remove […] My feeling is the refactoring necessary for this PR was “too big” [and genAI] **introduced as many errors as it fixed**.” Another developer comments that one prompt “failed to properly apply the edits and **started editing random other parts of the file**”

what happens when AI coders make iterative changes?

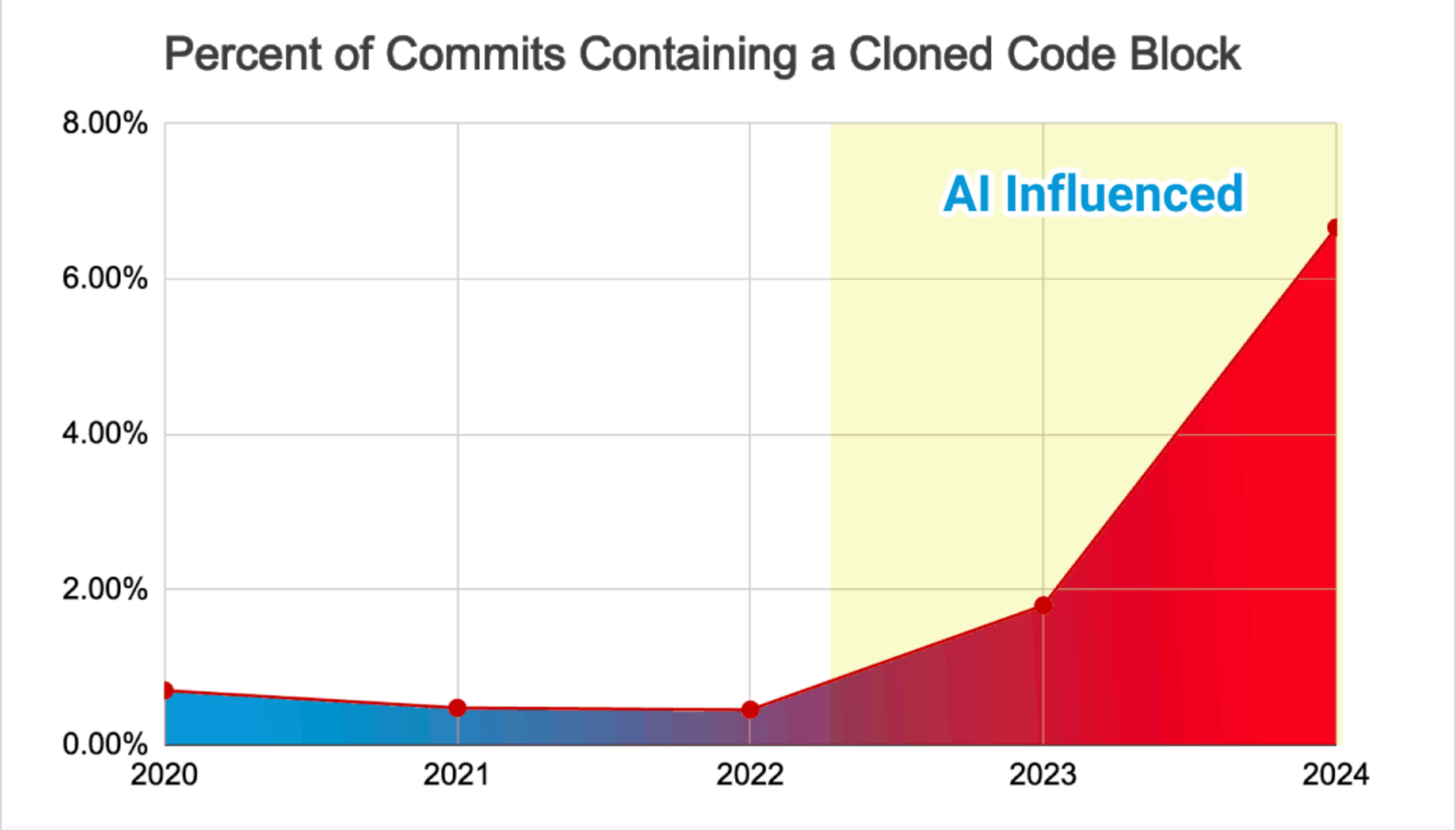


Gini coefficient against progress (0 means all functions same)

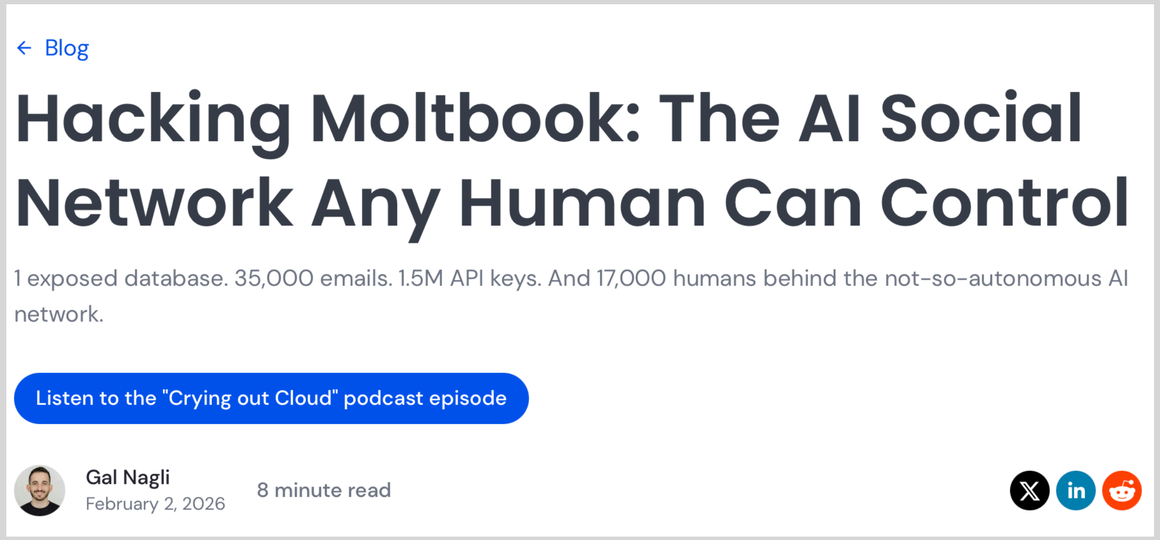
at each step, code becomes less balanced (and modular)

Slopcodebench (Orlanski et al, 2026)

what do commits look like?



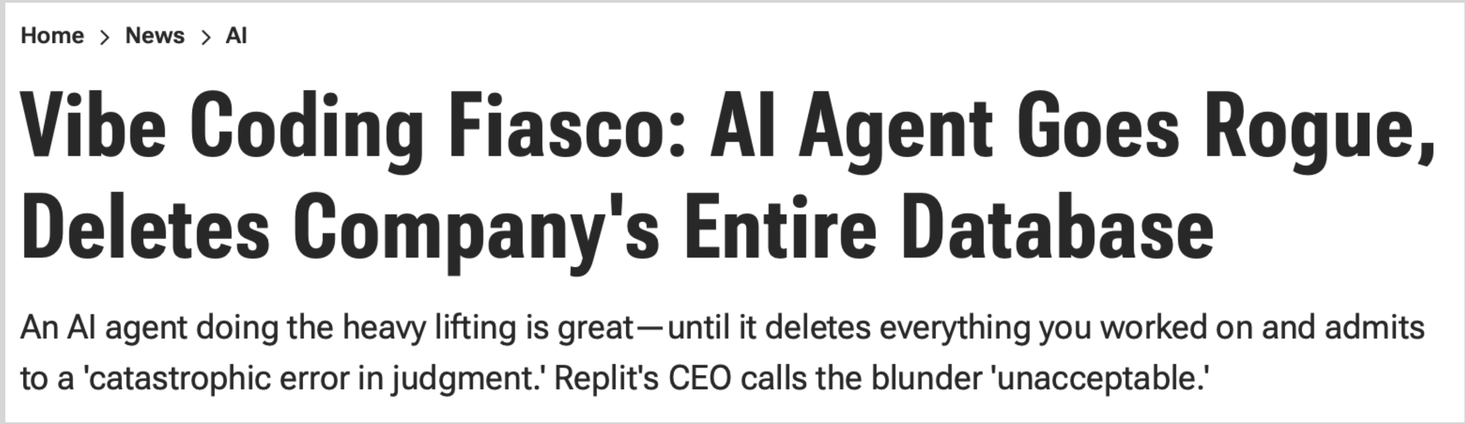
Gitclear study of >200m lines of code from Google, Microsoft, Meta et al over 5 years (2025)



Moltbook (February 2026)
database key in client-side JS
35k emails and 1.5m API keys exposed
also, no authentication of agents
mostly humans with huge fleets of bots

Remote Prompt Injection in GitLab Duo Leads to Source Code Theft

GitLab Duo (May 2025)
Claude-based coding assistant
prompt injection reveals private code



SaaStr (July 2025)
founder experimenting with vibecoding
Replit deleted production database

is LLM coding cost effective?

Claude Code incorrectly estimated API costs and executed expensive download without confirmation #16824

Closed as not planned

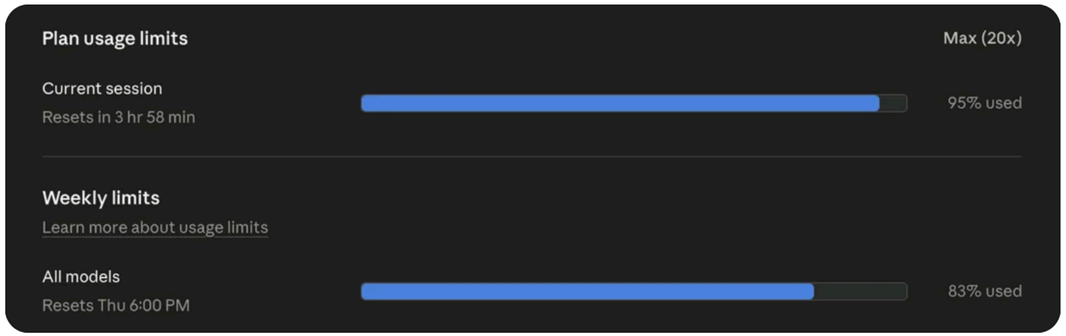
Claude bug (Jan 2026)
estimated $3 for task, charged $660

API Error

You've hit your limit · resets 4pm (America/New_York)

My own experience (April 2026)
one hour to make toy app
on first run, user registration fails
hit limit when asked Claude to fix

I pay $200/month for Claude Max and hit the limit in under 1 hour. What am I even paying for?



Claude limits
on Max plan ($200/mo) can reach in one hour
unpredictable, depends on time of day & load

Developer accidentally spends company's entire Cursor budget in one sitting — and discovers worrying flaw that let them extend it by over $1 million

OX Security says that a lack of mandatory spending caps and overly permissive access mean it's possible to drain corporate budgets within hours

Cursor design flaws (Dec 2025)
no proper spending caps or access control

where do the tokens go?

ACTIVITY	% OF TOKENS	COST CONTRIBUTION
File reading and code search	35-45%	**Largest single cost. Agent reads entire files when it needs one function.**
Tool/command output	15-25%	**CLI output, test results, error logs. 60 commands at 3,500 tokens each = 210K tokens of noise.**
Context re-sending	15-20%	Full conversation history resent on every API call. Grows linearly.
Reasoning and planning	10-15%	The agent thinking about what to do. Necessary but compounds with context size.
Code generation	**5-15%**	The part you actually want. The cheapest line item.

Morph (April 2026)
up to 80% of tokens wasted
due to poor context control

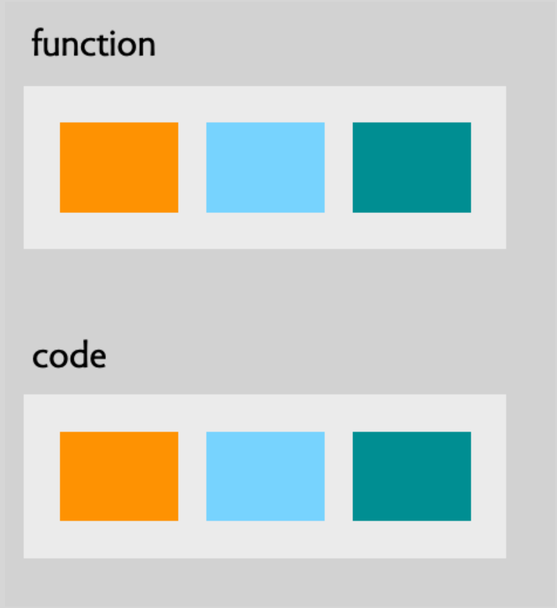
two ways
to work with AI

automate the mess	design for AI
take current practices as given	rethink our practices
let AI agents find structure	make expressive structures
outsource thinking to AI	outsource subtasks to AI
give AI unlimited context & power	focus AI intentionally

what might
design for AI
look like?

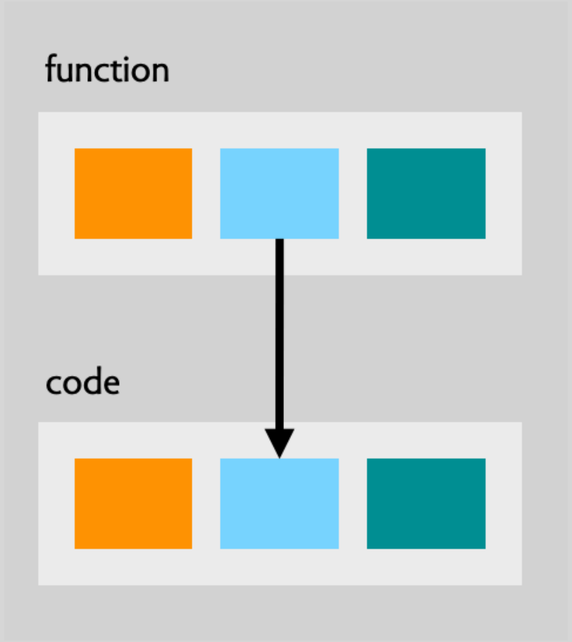
modularity

code structure = function structure
so easy to know where to make changes



context

context of LLM call tailored to task
no more or less than needed



legibility

code language = function language
so easy to translate between the two